HDF5: Past, Present, and Future

Quincey Koziol
Ruth Aydt
The HDF Group
{koziol,aydt}@hdfgroup.org

Condensed History of HDF5

1987: Graphics task force at NCSA began work on architecture-independent format and library, **HDF**.

1990: NSF provided funding to improve documentation, testing, and user support.

1994: NASA selected HDF as standard format for Earth Observing System.

1996–1998: DOE tri-labs and NCSA, with additional support from NASA, developed **HDF5**, initially called "BigHDF".

2005: NASA funded development of netCDF-4, a new version of netCDF that uses the HDF5 file format.

2006: **The HDF Group**, a non-profit corporation, spun off from NCSA and the University of Illinois.

HDF5 Technologies

- HDF5 Abstract Data Model
  - Groups, Datasets, Attributes, …

- HDF5 Software
  - Tools
  - High-level Libraries
  - Fortran, C++, Java Language Interfaces
  - HDF5 Library
    - C Language Interface
    - Internals
    - Virtual File Layer (I/O Drivers)

- HDF5 Binary File Format
  - Bit-level organization of stored data
  - Defined by HDF5 File Format Specification
    hdfgroup.org/HDF5/doc/H5.format.html

Who uses HDF5 Today?

- You!
- Academia, government agencies, industries around the world
- Over 200 different types of applications
  - Physical and life sciences; finance; manufacturing; film-making; engineering; decision-support; …
  - hdfgroup.org/HDF5/users5.html
- HDF5 technologies provide underpinnings for many formats / libraries / applications, such as:
  - netCDF-4
  - CFD General Notation System (CGNS) Mid-Level Library
  - NASA Earth Systems Data: HDF-EOS5
  - MATLAB® for datasets > 2GB

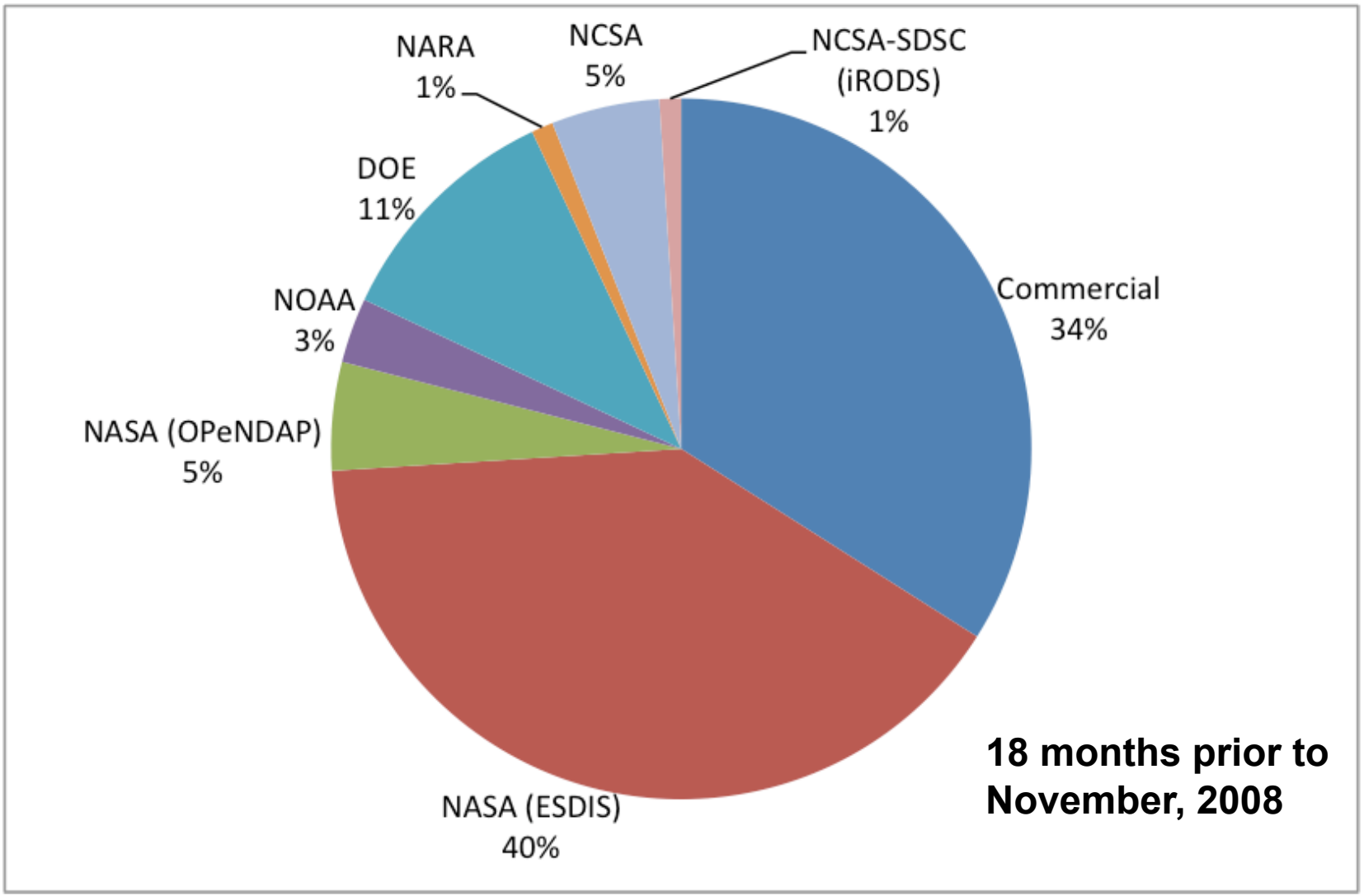
The HDF Group

- Non-profit 501(c)(3) corporation
- Owns and licenses HDF4 and HDF5
  - HDF formats and libraries to remain open
  - BSD-type license
  - University of Illinois receives royalties on income from commercial customers
- 20+ scientific, technology, professional staff

Mission:

To facilitate discovery, now and in the future, through stewardship of HDF technologies and support of HDF users.

Income Profile



Income Allocation

- Income apportioned to:
  - Direct work for customers
  - Ongoing stewardship of HDF
    - R&D, product and user support, …
  - Sustainability fund
  - Overhead expenses
    - office space and utilities, equipment and licensing fees, non-technical staff, benefits, …

Work for Customers

- Consult on use of HDF5 (domain data models, performance tuning, …)
  - NASA, NOAA, NCSA, Commercial, NIH*, DOD*
- Port to new platforms
  - NASA, Commercial
- Install, maintain, provide priority support
  - NASA, NOAA, NCSA, DOE, Commercial
- Perform custom development
  - NASA, DOE, Commercial, NIH*, DOD*
- Conduct training
  - NASA, NCSA, Commercial

** started after November 2008*

HPC-related* Income: SNL / NCSA

- Sandia NL: *Currently our only DOE-related funding*
  - Install and maintain on important platforms*
  - Priority support for HDF5 users
  - Special support for Java Interface and HDFView
  - Metadata Journaling (partial funding)
  - netCDF-4 (limited support)
- NCSA*: *Currently our only NSF-related funding*
  - Install, maintain, provide priority support on NCSA systems
  - Consult with application developers on use of HDF5
  - Parallel HDF5 Primer and Performance Tests
    - Currently under development
  - HDF5 / iRODS project
  - Tutorials / Talks

Product Support

- Software provided “as is” with no warranty of any kind.
  - hdfgroup.org/HDF5/doc/Copyright.html
- Software Portability
  - Want to support on “all” platforms [hardware, OS, compiler]
  - In reality, constrained by funding and access
  - Tested platforms and features published at:
    - hdfgroup.org/HDF5/release/platforms5.html
    - hdfgroup.org/HDF5/release/SuppConfigFeats5.html
    - Tested platforms include: (*not all features on all platforms*)
      - Systems: AIX, XT3, FreeBSD (i386, amd64), Irix (64 & n32), Linux SMP (i686, i386, x86_64, ia64, Intel 64, Altix ia64, Xeon w/ Lustre), SunOS (32 & 64), Windows (XP, XP x64, Vista), MacOS;
        Note: Cray XT4 and IBM BG are not tested
      - Compilers: xlc, xlf, xlC, mpcc_r, mpxlf_r, PGI (cc, f90, C++), gcc, g++, gfortran, g95, MIPSpro (cc, F90, C++), Intel (C, C++ Fortran), Absoft, Sun Workshop, Visual Studio, multiple MPI versions

Product Support

- Language Interfaces
  - Want to provide full capabilities from all languages
  - But, not there currently:
    - C APIs offer access to full set of library capabilities
    - Fortran, C++ more restricted; may be released after C
    - Java receives less attention; may be released after others
    - Interest in scripting languages, such as Python, but no substantial activity (in The HDF Group) at this time
- Assure long-term access to HDF5 data
  - Develop and adopt practices and technologies that allow HDF library to evolve, while maintaining data and software backward-compatibility.

User Support

- Documentation, Examples, Tutorials
  - Ongoing effort to improve these
- hdf-forum mailing list
  - Email hdf-forum-subscribe@hdfgroup.org to subscribe
- Helpdesk: help@hdfgroup.org
  - Available to all users; customers receive priority
  - Basic support, not a consulting service
  - Questions and comments help us identify bugs and shortcomings in code, examples, and documentation.

HDF5 Technical Road Map

Plans for Next Major Release

- *Performance* – Improvements “behind the scenes”
- *Ease of Use* - Fortran 2003 features
- *Robustness* – Metadata Journaling

Performance Improvements

- Improved metadata cache
- Rewrote file free-space manager:
  - Much faster
  - Persistent
- Added two chunk indexing methods: Fixed & Extensible Arrays
  - Faster
  - Single-Writer/Multi-Reader access to data

Fortran 2003 Features

- New Fortran 2003 features allow us to support
  - Any Fortran INTEGER and REAL type data in HDF5 files
  - Fortran derived types and HDF5 compound datatypes
  - Fortran enumerated types and HDF5 enumerated types
  - HDF5 APIs with callbacks

Metadata Journaling

- Problem:
  - Data in an open HDF5 file is susceptible to corruption in the event of an application or system crash.
- Initial Objective:
  - Guarantee an HDF5 file with consistent metadata can be reconstructed in the event of a crash; no guarantee on state of raw data.
- Approach:
  - For HDF5 API functions that modify metadata, a transaction is recorded in a journal file before the modified metadata is written to the HDF5 file.
  - If the application crashes, a recovery program can replay the journal and apply recorded metadata writes to the HDF5 file.
  - Design allows support for Parallel HDF5 to be added later.

HDF5 in the Future

"It's hard to make predictions, especially about the future" – Yogi Berra

Plans, Guesses, and Speculations

- Improve Multi-threaded Concurrency:
  - Currently thread-safe, but not concurrent
  - Start “pushing global lock down”
  - Use lock-free data structures inside library where possible
- Extend Single-Writer / Multiple-Reader capabilities:
  - Expand range of operations beyond new “extensible array” structure.
  - Support in Parallel HDF5

Plans, Guesses, and Speculations

- Improve Parallel I/O Performance:
  - Better leveraging of MPI and file system features
  - Reduce # of I/O operations for metadata access
  - Allow independent metadata creation operations
- Improve Robustness:
  - Journal raw data operations
  - Allow "super-transactions" to be created by applications
  - Support journaling for Parallel HDF5

Plans, Guesses, and Speculations

- Improve raw data chunk cache implementation
- Implement more efficient storage and I/O of variable-length data, including compression
- Work with HPC community to serve their needs:
  - Participating in MPI Forum
  - Focus on high-profile applications or “I/O cores” and address HDF5 bottlenecks
- Support remote access to HDF5 data
  - OPenDAP
  - iRODS
  - Web Services

Questions/Comments?